Robot Localization from Landmarks using Recursive Total Least Squares*

Daniel L. Boley and *Erik S. Steinmetz*

Department of Computer Science
University of Minnesota
Minneapolis, MN 55455

Karen T. Sutherland

Department of Computer Science
University of Wisconsin – La Crosse
La Crosse, WI 54601

Abstract

In the robot navigation problem, noisy sensor data must be filtered to obtain the best estimate of the robot position. We propose using a Recursive Total Least Squares algorithm to obtain estimates of the robot position. We avoid several weaknesses inherent in the use of the Kalman and extended Kalman filters, achieving much faster convergence without good initial (a priori) estimates of the position. The performance of the method is illustrated both by simulation and on an actual mobile robot with a camera.

1 Introduction

The purpose of this paper is to propose a simple scheme for estimating the position of a robot from relatively few sensor readings measuring some aspect of the environment. Our algorithms are intended for applications where sensor readings are expensive or otherwise limited so that only relatively few can obtained, and the readings that are taken are subject to considerable errors or noise. We propose a method capable of converging to a position estimate with greater accuracy using fewer measurments than other methods often used for this application, such as the Kalman and extended Kalman filter. Our approach is validated using a mobile robot on which a camera is used to obtain bearing information with respect to one or more landmarks in the environment.

The Kalman filter is often used when estimating the values of some dynamic quantity from noisy data [8], and also when trying to estimate a static quantity [12]. Its application to the robot navigation problem addressed in this paper was also discussed in [2]. Given the nonlinearity of the relationships between the bearings and the robot positions, often the extended Kalman filter is used [1, 10], by using a Taylor expansion to obtain a local linear approximation to the true relation. But the extended Kalman filter suffers from lack of robustness. It can often fail to converge entirely [13]. This led us to develop a linear formulation of the estimation problem that is not a local approximation, but holds through the entire range of parameter values.

Although limited modifications can be made to the Kalman approach to improve robustness to noise [11], our work in outdoor navigation [16], where measurements are expensive to obtain and have significant error inherent to the system, motivated us to look for another filtering method, preferably one which would not require numerous measurements to converge and did not assume an error-free data matrix. As demonstrated by Mintz et al. [7], the sense in which a method is said to be "optimal" depends critically on the specific model being used. When error exists in both the measurement and the data matrices, the best solution in the *least squares* sense is often not as good as the best solution in the *eigenvector* sense. This second method is known in the statistical literature as *orthogonal regression* and in numerical analysis as *total least squares* (TLS) [17].

To demonstrate the algorithms, we use a mobile robot platform on which is mounted a camera. The sensor readings are obtained by viewing one or more landmarks in the visual images obtained as the robot moves. The images yield several bearings to the landmarks, which are then used to estimate the position of the robot. The bearings are subject to considerable noise, both from the coarseness of the image resolution and from odometry error in fixing the base line. In spite of the noise in the data, location of the robot could be fixed with relatively high accuracy.

In this paper, we show how the task of estimating

*This work was supported jointly by Minnesota Department of Transportation grant 71789-72996-173 and National Science Foundation grant CCR-9405380.

robot positions from bearing data can be formulated directly as a simple, linear matrix problem, which is not a local linearized approximation, but is valid globally. We propose using a Total Least Squares approach, which has the advantage over the Kalman filter of admitting errors anywhere in the equations. This paper is organized as follows. After this introduction, we discuss the Recursive TLS algorithm in section 2, our experimental results in section 3, and some concluding remarks in section 4.

2 Recursive Total Least Squares Algorithm

Given an overdetermined system of equations $A\mathbf{x} = \mathbf{b}$, the TLS problem, in its simplest form, is to find the smallest perturbation to A and $\mathbf{b}$ to make the system of equations compatible. Specifically, we seek a matrix E and vector $\mathbf{f}$ that minimizes $\|(E, \mathbf{f})\|_2$ such that $(A+E)\mathbf{x} = \mathbf{b} + \mathbf{f}$ for some vector $\mathbf{x}$. The vector $\mathbf{x}$ corresponding to the optimal $(E, \mathbf{f})$ is called the *TLS solution*. Recently, some recursive TLS filters have been developed for applications in signal processing [4, 5, 19]. Davila [4] used a Kalman filter to obtain a fast update for the eigenvector corresponding to the smallest eigenvalue of the covariance matrix. This eigenvector was then used to solve a symmetric TLS problem for the filter. It was not explained how the algorithm might be modified for the case where the smallest eigenvalue is multiple (i.e., corresponding to a noise subspace of dimension higher than one), or variable (i.e., of unknown multiplicity). In [19], Yu described a method for the fast update of an approximate eigendecomposition of a covariance matrix. He replaced all the eigenvalues in the noise subspace with their "average", and did the same for the eigenvalues in the signal subspace, obtaining an approximation which would be accurate if the exact eigenvalues could be grouped into two clusters of known dimensions. In [5], DeGroat used this approach combined with the averaging technique used in [19], again assuming that the singular values could be grouped into two clusters. Recently, Bose et al.[3] applied Davila's algorithm to reconstruct images from noisy, undersampled frames after converting complex-valued image data into equivalent real data. All of these methods made some assumptions that the singular values or eigenvalues could be well approximated by two tight clusters, one big and one small. In this paper, we present a recursive algorithm that makes very few assumptions about the distribution of the singular values.

The most common algorithms to compute the TLS solution are based on the Singular Value Decomposition (SVD), a non-recursive matrix decomposition which is computationally expensive to update. The TLS problem can be solved by the SVD using Algorithm 3.1 of [17]. The main computation cost of that algorithm occurs in the computation of the SVD. That cost is $O(p^3)$ for each update. The basic solution method is sketched as follows. If $\mathbf{v} = (v_1, \ldots, v_p)^T$ is a right singular vector corresponding to the smallest singular value of $(A, \mathbf{b})$, then it is well known that the TLS solution can be obtained by setting $\mathbf{x} = -(v_1, \ldots, v_{p-1})^T / v_p$. If the smallest singular value is multiple, then there are multiple TLS solutions, in which case one usually seeks the solution of smallest norm. If v_p is too small or zero, then the TLS solution may be too big or nonexistent, in which case an approximate solution of reasonable size can be obtained by using the next smallest singular values(s) [17].

In cases such as the applications considered in this paper where the exact TLS solution is still corrupted by external effects such as noise, it suffices to obtain an approximate TLS solution at much less cost. We seek a method that can obtain a good approximation to the TLS solution, but which admits rapid updating as new data samples arrive. One such method is the so-called ULV Decomposition, first introduced by Stewart [14] as a means to obtain an approximate SVD which can be easily updated as new data arrives, without making any a priori assumptions about the overall distribution of the singular values. The ULV Decomposition of a real $n \times p$ matrix A (where $n \geq p$) is a triple of 3 matrices U, L, V plus a rank index r, where $A = ULV^T$, V is $p \times p$ and orthogonal, L is $p \times p$ and lower triangular, U has the same shape as A with orthonormal columns, and where L has the form

$$L = \begin{pmatrix} C & 0 \\ E & F \end{pmatrix}$$

where C $(r \times r)$ encapsulates the "large" singular values of A, (E, F) $((p-r) \times p)$ approximately encapsulate the $p - r$ smallest singular values of A, and the last $p - r$ columns of V encapsulate the corresponding trailing right singular vectors. To solve the TLS problem, the U matrix is not required, hence we need to carry only L, V, and the effective rank r. Therefore, a given ULV Decomposition can be represented just by the triple $[L, V, r]$.

The feature that makes this decomposition of interest is the fact that, when a new row of coefficients is appended to the A matrix, the L, V and r can be updated in only $O(p^2)$ operations, with L restored to the standard form above, as opposed to the $O(p^3)$ cost for an SVD. In this way, it is possible to track the leading r-dimensional "signal subspace" or the remaining "noise subspace" relatively cheaply. Details on the updating process can be found in [14, 9].

We can adapt the ULV Decomposition to solve the Total Least Squares (TLS) problem $A\mathbf{x} \approx \mathbf{b}$, where new measurements b are continually being added, as proposed in [2]. The adaptation of the ULV to the TLS problem has also been analyzed independently in great detail in [18], though the recursive updating process was not discussed at length. For our specific purposes, let A be an $n \times (p-1)$ matrix and $\mathbf{b}$ be an n-vector, where p is fixed and n is growing as new measurements arrive. Then given a ULV Decomposition of the matrix $(A, \mathbf{b})$ and an approximate TLS solution to $A\mathbf{x} \approx \mathbf{b}$, our task is to find a TLS solution $\widehat{\mathbf{x}}$ to the augmented system $\widehat{A}\widehat{\mathbf{x}} \approx \widehat{\mathbf{b}}$, where

$$\widehat{A} = \begin{pmatrix} \lambda A \\ \mathbf{a}^T \end{pmatrix} \text{ and } \widehat{\mathbf{b}} = \begin{pmatrix} \lambda \mathbf{b} \\ \beta \end{pmatrix},$$

and λ is an optional exponential forgetting factor [8].

The RTLS Algorithm:

- Start with $[L, V, r]$, the ULV Decomposition of $(A, \mathbf{b})$, and the coefficients $\mathbf{a}^T, \beta$ for the new incoming equation $\mathbf{a}^T\mathbf{x} = \beta$.

- Compute the updated ULV Decomposition for the system augmented with the new incoming equation. Denote the new decomposition by $[\widehat{L}, \widehat{V}, \widehat{r}]$.

- Partition
$$\widehat{V} = \begin{pmatrix} \widehat{V}_{11} & \widehat{V}_{12} \\ \widehat{V}_{21} & \widehat{V}_{22} \end{pmatrix},$$
where $\widehat{V}_{22}$ is $1 \times (p - \widehat{r})$.

  If $\|\widehat{V}_{22}\|$ is too close to zero (according to a user supplied tolerance), then we can adjust the rank boundary $\widehat{r}$ down to obtain a more robust, but approximate solution [2, 9].

- Find an orthogonal matrix Q such that $\widehat{V}_{22}Q = (0, \ldots, 0, \alpha)$, and let $\mathbf{v}$ be the last column of $\widehat{V}_{12}Q$. Then compute the new approximate TLS solution according to the formula $\widehat{\mathbf{x}} = -\mathbf{v}/\alpha$.

This RTLS Algorithm makes very few assumptions about the underlying system, though the user must supply a zero tolerance and a gap tolerance for $\|\widehat{V}_{22}\|$. For the application here, it sufficed to set the zero tolerance to zero and depend on just the gap tolerance of 1.5.

3 Experimental Results

To compare the performance of the Kalman filter and RTLS in practice, we ran two sets of experiments, the first with a physical mobile robot and camera and a single landmark, and the second in simulation with two landmarks. The setup in the first set was modeled after the problems faced by an actual mobile robot [1, 6, 10]. The robot did not know its own position on the map, but did know the location of a single landmark. The robot moved in a straight line taking a series of images. Its task was to find the landmark in each image, and use the results to determine its start position relative to the landmark.

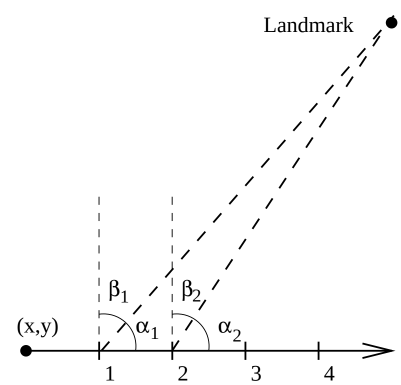

Figure 1: *Diagram illustrating angles to landmark. TRC Labmate had camera mounted at* $90°$*, yielding bearing* β*, which was bounded by* $\pm 25°22'$ *for the given field of view.*

A Panasonic WV-BL202 camera was mounted on a TRC Labmate at an angle of $90°$ to robot bearing, so that each image yields an angle β_i, as shown in Figure 1. Horizontal field of view was $50°44'$, limiting the angles β to the range $\pm 25°22'$. "Landmarks" were mini Maglite high intensity flashlight candles. The angular position of the landmark was measured in a sequence of images taken while the robot moved across the room at a constant velocity. In addition to the error in angle measure, error also occurred in velocity, robot bearing and in the times at which the images were taken. It is not possible to predict and model these errors. For example, velocity was set at 20mm/second, but average true velocity across runs ranged from 21.4mm/second to 22.5mm/second. In addition, the supposed constant velocity was not constant throughout a single run, varying in an unpredictable manner. It would be unrealistic to assume any of these errors or their combined result to have a gaussian distribution.

It is assumed that the landmark is located at (0,0),

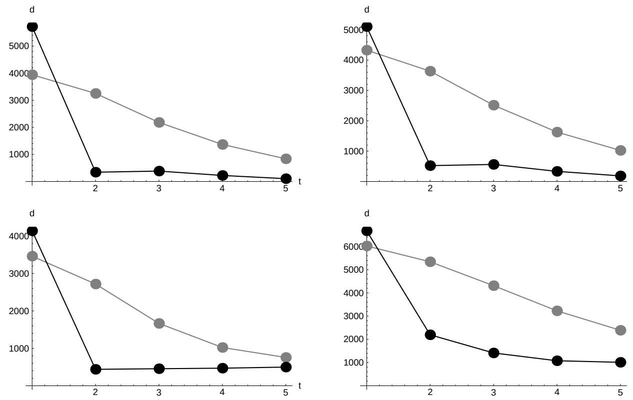

Figure 2: *Performance of RTLS (black) and Kalman filter (grey) on runs using the TRC Labmate starting with 4 different landmark locations. Images were grabbed at time intervals t (horizontal axis) 12 seconds apart. The vertical axis gives the deviation of the estimated start position from the actual start position in millimeters.*

that the y coordinate of the robot's position does not change as the robot moves, and that the robot knows which side of the landmark it is on. At any step i:

$$tan(\beta_i) = \frac{x + (t_0 + i * interval) * velocity}{y}$$

where (x, y) is the robot start position, β_i is the measured angle, t_0 is robot start time, *interval* is the interval at which images are grabbed and *velocity* is the robot velocity. The problem was expressed as a linear function so that no accuracy was lost by linearizing. However, the data matrix as well as the measurement vector contained error:

$$A_i = \left[\begin{array}{cc} 1 & -tan(\beta_i) \end{array}\right]; \quad \mathbf{x}_i = \left[\begin{array}{c} x \\ y \end{array}\right];$$
$$\mathbf{b}_i = -(t_0 + i * interval) * velocity$$

where at any step i, A_i is the data matrix, $\mathbf{b}_i$ is the measurement vector and $\mathbf{x}_i$ is the estimated state vector consisting of the coordinates (x, y) of the robot start position. The Kalman filter was given an estimated start position of (0,0), so that the deviation at time 0 for the Kalman filter is just the initial distance from the robot to the landmark. The leading column of the data matrix was weighted by $\eta = 100$ (to account for the fact that this column has no error).

Figure 2 shows a comparison of four of the robot runs. The robot velocity was set to 20mm/sec. Five images were grabbed 12 seconds apart. The robot start position relative to the landmark used for localization was different in each run. The deviations d of the estimate of start location from actual start location at each 12 second time interval t are compared. The RTLS filter converged faster and to more accuracy than did the Kalman, often requiring only 2 or 3 steps to achieve full accuracy.

The second set of experiments was run in simulation, but used two landmarks without assuming any prior knowledge of the robot's heading. We assume that the robot has no instrument such as a compass which could be used to register its compass heading. Such instruments can give varying, incorrect readings in outdoor, unstructured environments [16], so that it is useful to design and evaluate methods to obtain heading information from external sources. Such heading information could be used independently or as corrections to estimates from internal sources. The robot knows the location of the two landmarks on a map (ground coordinate system). A coordinate system is arbitrarily centered at one landmark. The goal is to determine the robot start position plus the location of the second landmark. Knowing which landmark is which in the view will allow the robot to uniquely determine its starting position from multiple readings along a baseline of unknown direction, except for certain degenerate configurations. Even if the robot does not know the order of the two landmarks in its view, it can limit its start position to only two possible locations in the ground coordinate system, symmetrically located on either side of the line joining the landmarks, without any *a priori* knowledge of direction.

The robot coordinate system is defined by placing landmark 1 at $(0, 0)$ and landmark 2 at coordinates (l, m) to be determined by the filter. The x-axis is defined by the direction of the robot heading. The computed coordinates (l, m) permit mapping this coordinate system to the ground coordinate system. We let α_{1i}, α_{2i} be the angles from the robot heading to each of the landmarks at time t_i. This is illustrated for each individual landmark in Fig. 1. We have the following relationships:

$$-sin(\alpha_{1i}) * x + cos(\alpha_{1i}) * y = t_i * velocity * sin(\alpha_{1i})$$
$$-sin(\alpha_{2i}) * (x - l) + cos(\alpha_{2i}) * (y - m) = t_i * velocity * sin(\alpha_{2i})$$

where (x, y) is the robot start position. Random error with a uniform distribution was added to the angle measures and a normally distributed random error was added to the time measurement. As in the previous experiments, the problem was expressed as a linear function with the data matrix as well as the measurement vector containing error:

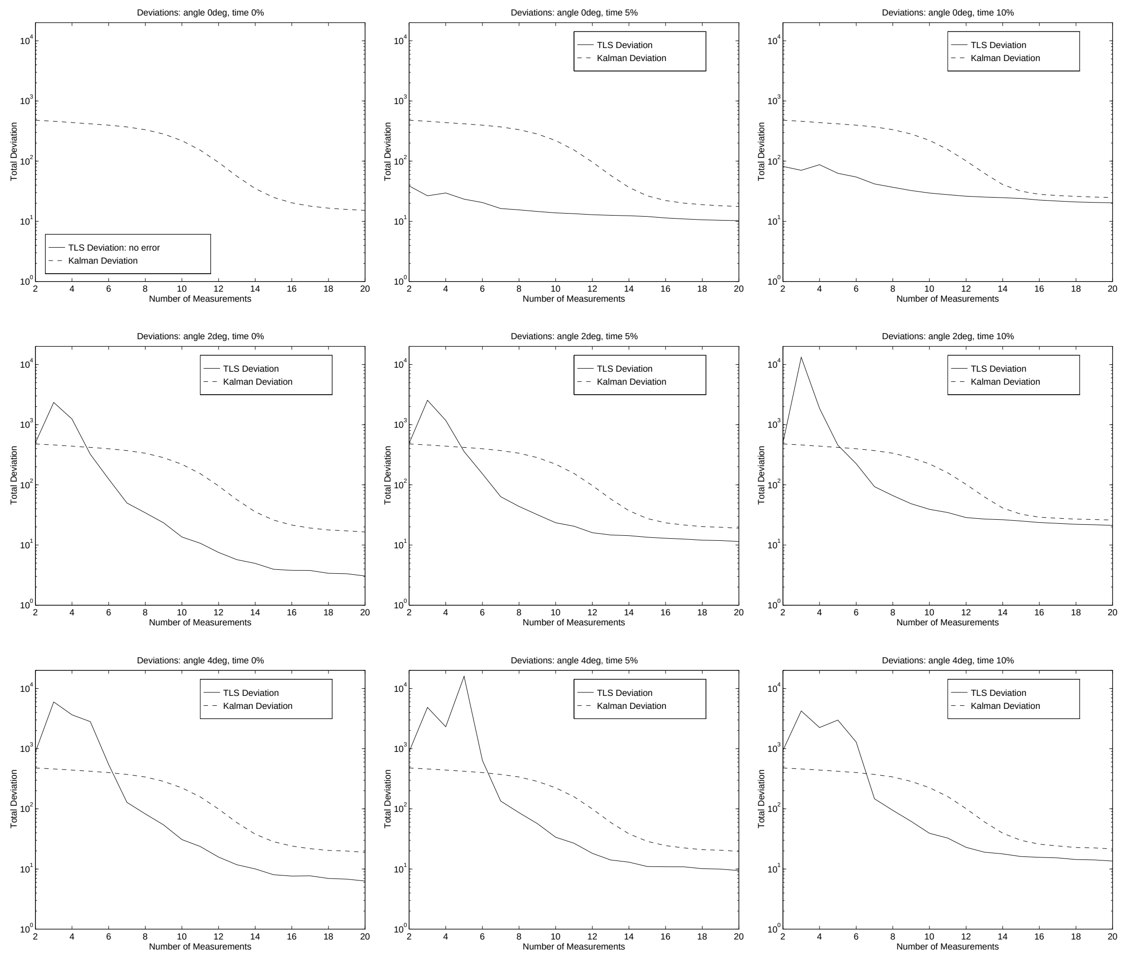

Figure 3: *Mean deviations (d on vertical axis) between estimated and actual start positions, versus time steps (t on horizontal axis). Each row of plots shows the results with uniform errors in the angles of* $0, \pm 2°, \pm 4°$*, respectively, and each column shows the results with normally distributed errors in t with standard deviations* $0, 5\%, 10\%$*, respectively.*

$$A_i = \begin{bmatrix} -sin(\alpha_{1i}) & cos(\alpha_{1i}) & 0 & 0 \\ -sin(\alpha_{2i}) & cos(\alpha_{2i}) & sin(\alpha_{2i}) & -cos(\alpha_{2i}) \end{bmatrix};$$

$$\mathbf{x}_i = \begin{bmatrix} x \\ y \\ l \\ m \end{bmatrix}; \quad \mathbf{b}_i = \begin{bmatrix} t_i * velocity * sin(\alpha_{1i}) \\ t_i * velocity * sin(\alpha_{2i}) \end{bmatrix}$$

where at any step i, A_i is the data matrix, $\mathbf{b}_i$ is the measurement vector and $\mathbf{x}_i$ is the estimated state vector consisting of the coordinates (x, y) of the robot start position and the coordinates (l, m) of the second landmark. Figure 3 summarizes the results in an example where the two landmarks and the robot were placed at positions $(-200, 0)$, $(0, 0)$, and $(-200, -200)$, respectively, in the ground coordinate system. When the angle error is negligible, the TLS method provides uniformly good estimates. When the angle error is moderate, the error from TLS method suffers from an initial jump, but quickly recovers because it needs no initial estimate. Furthermore, in the regions where the RTLS error exceeds the Kalman filter error, neither filter yields any accuracy at all, since both errors are larger than the values being estimated.

4 Conclusion

In this paper, we have proposed a Recursive Total Least Squares (RTLS) filter. This filter is easily updated as new data arrives, yet makes very few assumptions about the data or the problem being solved. The method was based on the ULV Decomposition. We have suggested its use as an alternative to the

Kalman filter in reducing uncertainty in robot navigation. In this context RTLS does not require an initial state estimate, avoids modeling errors introduced by the extended Kalman filter, does not suffer the traps of local minima, and converges quickly. We have illustrated the method with simulated as well as actual robot runs. It is demonstrated that in the domain of robot navigation the RTLS can often provide more accurate estimates in fewer time steps than the Kalman filter, especially when errors are present in both the measurement vector and the data matrix. Future work includes utilizing the filter in navigation problems with actual outdoor terrain data and combining its use with the higher level reasoning described in [15].

References

[1] N. Ayache and O. D. Faugeras. Maintaining representations of the environment of a mobile robot. *IEEE Trans. Robot. & Auto.*, 5(6):804–819, December 1989.

[2] D. L. Boley, E. S. Steinmetz, and K. T. Sutherland. Recursive total least squares: An alternative to the discrete Kalman filter in robot navigation. Technical Report CS TR 95-049, University of Minnesota, July 1995.

[3] N. K. Bose, H. C. Kim, and H. M. Valenzuela. Recursive implementation of total least squares algorithm for image reconstruction from noisy, undersampled multiframes. In *Proceedings 1993 Intl. Conf. on Acoustics, Speech and Signal Processing*, pp. V–269–V–272. IEEE, May 1993.

[4] C. E. Davila. Efficient recursive total least squares algorithm for FIR adaptive filtering. *IEEE Trans. Sig. Proc.*, 42(2):268–280, 1994.

[5] R. D. DeGroat. Noniterative subspace tracking. *IEEE Trans. Sig. Proc.*, 40(3):571–577, March 1992.

[6] H. Durrant-White, E. Bell, and P. Avery. The design of a radar-based navigation system for large outdoor vehicles. In *Proceedings 1995 Intl. Conf. on Robot. & Auto.*, pp. 764–769. IEEE, June 1995.

[7] G. Hager and M. Mintz. Computational methods for task-directed sensor data fusion and sensor planning. *The Intl. Journal of Robotics Research*, 10(4):285–313, August 1991.

[8] S. Haykin. *Adaptive Filter Theory.* Prentice Hall, 2nd edition, 1991.

[9] S. Hosur, A. H. Tewfik, and D. Boley. Multiple subspace ULV algorithm and LMS tracking. In M. Moonen and B. D. Moor, editors, *3rd Int'l Workshop on SVD and Signal Processing*, pp. 295–302. Elsevier, August 1995. Leuven, Belgium.

[10] A. Kosaka and A. C. Kak. Fast vision-guided mobile robot navigation using model- based reasoning and prediction of uncertainties. *CVGIP: Image Understanding*, 56(3):271–329, Nov. 1992.

[11] H. Schneiderman and M. Nashman. A discriminating feature tracker for vision-based autonomous driving. *IEEE Trans. Robot. & Auto.*, 10(6):769–775, December 1994.

[12] R. C. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainty. *The Intl. Journal of Robotics Research*, 5(4):56–68, Winter 1986.

[13] H. W. Sorenson. Least-squares estimation: from Gauss to Kalman. *IEEE Spectrum*, pp. 63–68, July 1970.

[14] G. W. Stewart. Updating a rank-revealing ULV decomposition. *SIAM J. Matrix Analysis*, 14(2), 1993.

[15] K. T. Sutherland. Ordering landmarks in a view. In *Proceedings 1994 ARPA Image Understanding Workshop*, Nov. 1994.

[16] K. T. Sutherland and W. B. Thompson. Localizing in unstructured environments: Dealing with the errors. *IEEE Trans. Robot. & Auto.*, 10(6):740–754, December 1994.

[17] S. Van Huffel and J. Vandewalle. *The Total Least Squares Problem - Computational Aspects and Analysis.* SIAM, Philadelphia, 1991.

[18] S. Van Huffel and H. Zha. An efficient total least squares algorithm based on a rank-revealing two-sided orthogonal decomposition. *Numerical Algorithms*, 4(1-2):101–133, January 1993.

[19] K.-B. Yu. Recursive updating the eigenvalue decomposition of a covariance matrix. *IEEE Trans. Sig. Proc.*, 39(5):1136–1145, May 1991.